

# Graph CNN for Moving Object Detection in Complex Environments from Unseen Videos

Jhony H. Giraldo<sup>1</sup>, Sajid Javed<sup>2</sup>, Naoufel Werghi<sup>2</sup>, Thierry Bouwmans<sup>1</sup>

<sup>1</sup>La Rochelle Université, <sup>2</sup>Khalifa University of Science and Technology

{jgiral01, tbouwman}@univ-lr.fr, {sajid.javed, naoufel.werghi}@ku.ac.ae

## Abstract

*Moving Object Detection (MOD) is a fundamental step for many computer vision applications. MOD becomes very challenging when a video sequence captured from a static or moving camera suffers from the challenges: camouflage, shadow, dynamic backgrounds, and lighting variations, to name a few. Deep learning methods have been successfully applied to address MOD with competitive performance. However, in order to handle the overfitting problem, deep learning methods require a large amount of labeled data which is a laborious task as exhaustive annotations are always not available. Moreover, some MOD deep learning methods show performance degradation in the presence of unseen video sequences because the testing and training splits of the same sequences are involved during the network learning process. In this work, we pose the problem of MOD as a node classification problem using Graph Convolutional Neural Networks (GCNNs). Our algorithm, dubbed as GraphMOD-Net, encompasses instance segmentation, background initialization, feature extraction, and graph construction. GraphMOD-Net is tested on unseen videos and outperforms state-of-the-art methods in unsupervised, semi-supervised, and supervised learning in several challenges of the Change Detection 2014 (CDNet2014) and UCSD background subtraction datasets.*

## 1. Introduction

Moving Object Detection (MOD) is a crucial problem in computer vision for applications such as video surveillance, pose estimation, and intelligent visual observation of animals, to name just a few [18, 14]. MOD mainly aims to separate the moving objects known as foreground from the static component known as background [5]. In the literature, MOD has been considered as a binary classification problem where each pixel is predicted for either background or foreground component in a sequence taken from a static or moving camera. Therefore, this problem is also known

as background-foreground segregation [7]. MOD becomes a very challenging problem because of the presence of dynamic variations in the background scene such as illumination variations, swaying bushes, camouflage, intermittent object motion, shadows, and jittering effects, to name a few [49]. In addition, videos that are taken from PanTiltZoom (PTZ) and moving cameras also pose more challenges for efficient MOD. Several methods have been proposed to improve the performance of MOD under challenging scenarios such as those presented in [36, 40, 17]. However, there is no unique unsupervised or supervised method that can effectively handle all the aforementioned challenges in real scenarios [48, 49].

Many state-of-the-art MOD methods such as SuB-SENSE [46] work well for sequences taken from a static camera; however, they show performance degradation in the presence of moving camera sequences as shown in Fig. 1. And in fact, none of the methods proposed so far can address all challenges posed by static as well as moving cameras. In the current work, we contribute to bridging these gaps by formulating the MOD problem on Graph Convolutional Neural Networks (GCNNs). We propose a novel semi-supervised algorithm dubbed Graph MOD Network (GraphMOD-Net) based on GCNNs [24]. GraphMOD-Net models the instances in videos as nodes embedded in a graph. The instances are obtained with a Mask Region-CNN (Mask R-CNN) [22] or Cascade Mask R-CNN [10]. The representation of the nodes is obtained with background initialization, optical flow, intensity, and texture features [30, 35]. Moreover, these nodes are associated either with the class foreground or background using ground-truth information and finally, a GCNN is trained with a small percentage of labeled nodes to perform a semi-supervised learning classification [24] with an unseen scheme [48]. GraphMOD-Net outperforms state-of-the-art methods in several challenges of the Change Detection 2014 (CDNet2014) [49] and UCSD background subtraction [31] datasets. The main contributions of this paper are summarized as follows:

- We pose the problem of MOD as a binary classification

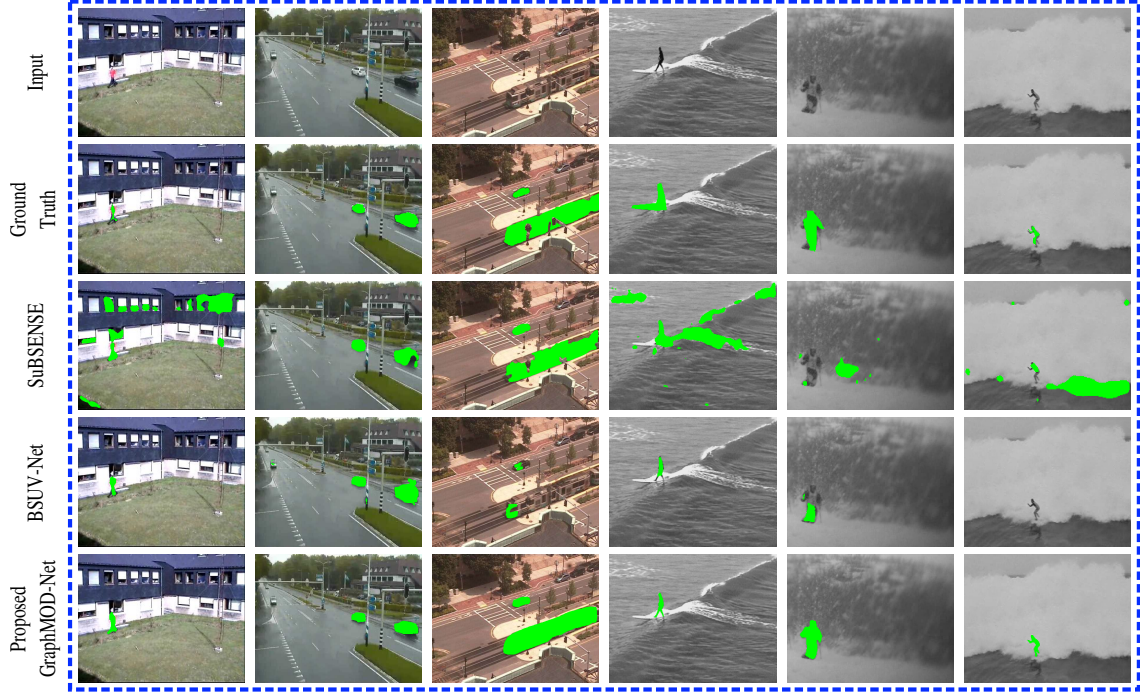


Figure 1. Comparisons of the visual results of the proposed GraphMOD-Net algorithm with two state-of-the-art methods for MOD on six very challenging video sequences taken from CDNet2014 [49] and UCSD [31] datasets, including moving camera sequences plus dynamic backgrounds. From left to right: zoom in zoom out, two-position PTZ cameras, intermittent pan, surfers, skiing, and surf sequences. From top to bottom: input frame, the ground truth of MOD, SuBSENSE [46], BSUV-Net [48], and our proposed GraphMOD-Net.

problem on the graph where each node is classified into two distinct components including background and foreground using GCNNs.

- We perform rigorous experiments using the CDNet2014 dataset [49] for MOD. Our results demonstrate that GraphMOD-Net uses a limited amount of labeled data and outperforms some state-of-the-art classical and deep learning methods.

The rest of the paper is organized as follows. Section 2 presents the related works. Section 3 explains the basic concepts and the proposed GraphMOD-Net. Section 4 introduces the experimental framework. And finally, Sections 5 and 6 present the results and conclusions, respectively.

## 2. Related Work

The MOD methods can be categorized into unsupervised, semi-supervised, and supervised learning methods. The unsupervised methods can be classified as statistical [6, 34, 54], subspace learning [4], and robust principal component analysis models [9, 41]. These methods do not successfully adapt to complex scenarios in MOD. Inspired by the success of deep CNNs on a wide variety of visual recognition tasks [42], several studies have also been proposed to handle the MOD problem in a fully supervised manner [15, 8]. However, these deep learning methods usually re-

quire a large amount of training data. The annotation of such data is an intensive task and can be very expensive to obtain the required degree of quality. Even though some works have tried to explain the success of deep learning methods for MOD [32], there is no concrete evidence in the literature about the sample complexity required in the deep learning regimen [13]. Furthermore, being trained and tested on sequences derived from the same videos, most of the deep learning methods for MOD do not show good generalization to unseen videos exhibiting unpredictable changes. The performance degradation was confirmed in CDNet2014 (FgSegNetv2 [26]) for which the performance of leading methods dropped dramatically when evaluated on unseen videos as reported in [48]. For a complete review of supervised and unsupervised methods, we invite the readers to the survey papers [6, 5, 9, 8].

Giraldo and Bouwmans [16] have recently proposed a Graph-based semi-supervised learning method for Background Subtraction (GraphBGS). Leveraging the theory of sampling and graph signal reconstruction, this framework found applications in MOD [37]. GraphBGS exploits a variational approach to solve the semi-supervised learning problem [39], assuming that the underlying signals corresponding to the background/foreground nodes are smooth in the graph [11]. Therefore, this method shows performance degradation if the smooth prior assumption is not fully sat-

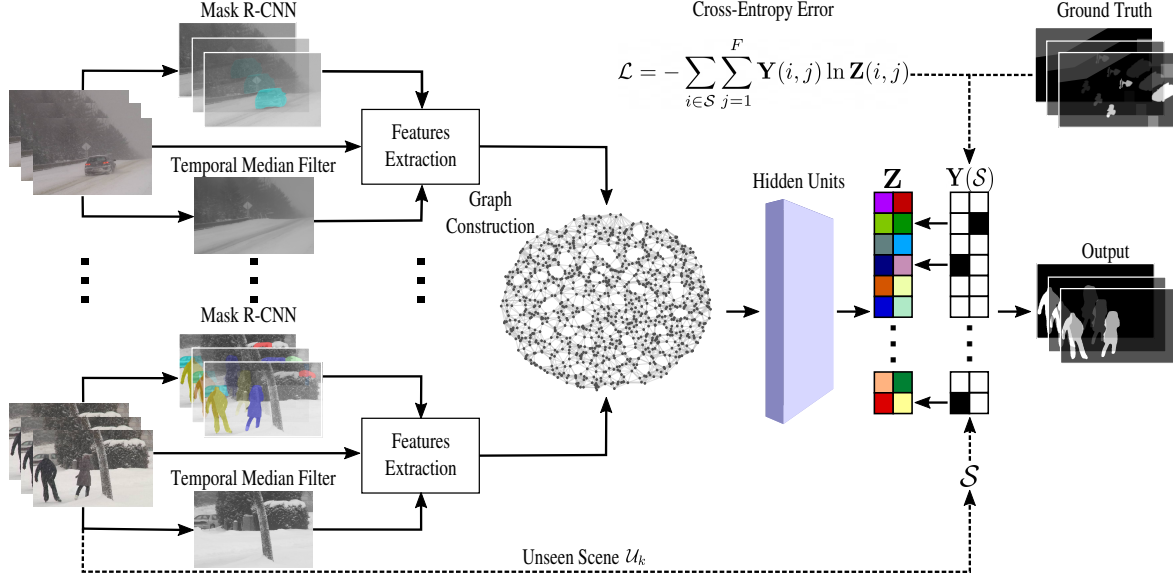


Figure 2. GraphMOD-Net uses background initialization and instance segmentation. Each instance represents a node in a graph using motion, intensity, and texture features. Finally, a GCNN classifies if each node is a moving or static object with an unseen scheme.

ified. In this work, we propose a semi-supervised learning algorithm dubbed as GraphMOD-Net. Our algorithm extends the work of Giraldo and Bouwmans [16] by increasing modeling capacity and avoiding the explicit smoothness assumption with GCNNs. GraphMOD-Net has the advantage of requiring less training data than deep learning methods while adapting to complex MOD scenarios, unlike unsupervised methods.

### 3. Proposed Algorithm

Our proposed algorithm consists of several components including instance segmentation, background model initialization, features extraction, graph construction, and GCNN training, as shown in the block diagram in Fig. 2. In the first step, we compute the instance segmentation mask of the input sequence using the Mask R-CNN method [22]. The instance segmentation mask assists our algorithm to get the prior knowledge of the object instances in the training videos. We also compute the initial background model using the temporal median filtering method. In the second step, the instances and background model are used to derive the features to be used in the graph construction. We use texture features, motion features using optical flow estimation [30], and intensity features. However, other types of features can be used to represent the nodes in the graph [17]. In the second step, we construct the graph where a node is assigned to each object instance represented by the group of the aforementioned features. With the so obtained graph representation, we train a GCNN to classify the nodes into either background or foreground. The different components of the algorithm will be described next.

#### 3.1. Instance Segmentation Description

We used the Mask R-CNN [22], with a Residual Network of 50 layers (ResNet50) [22] and Feature Pyramid Network (FPN) [28] as the backbone, for instance segmentation in the CDNet2014. While for the UCSD dataset we used Cascade Mask R-CNN [10] with ResNeSt of 200 layers [51]. Both instance segmentation networks are trained with the Common Objects in Context (COCO) 2017 dataset [27]. Each output (mask) of the instance segmentation method represents a node in our graph. We note that super-pixels can be used as an alternative for object instances in GraphMOD-Net. However, super-pixels and pixel-based approaches, in general, produce a huge-sized graph, making the computation cost-prohibitive and the implementation impractical [38, 29].

#### 3.2. Graph Construction

A graph  $G$  can be represented with two sets as  $(\mathcal{V}, \mathcal{E})$ , where  $\mathcal{V} = \{1, \dots, N\}$  is the set of nodes, with cardinality  $|\mathcal{V}| = N$  is the number of nodes of  $G$ .  $\mathcal{E} = \{(i, j)\}$  is the set of edges, where  $(i, j)$  represents an edge between the nodes  $i$  and  $j$ . The adjacency matrix of  $G$  is  $\mathbf{A} \in \mathbb{R}^{N \times N}$ , where  $\mathbf{A}(i, j)$  is the weight of the edge  $(i, j)$ . A graph is called unweighted when  $\mathbf{A}(i, j) = 1 \forall (i, j) \in \mathcal{E}$ . For undirected graphs  $\mathbf{A}$  is symmetric, *i.e.*, the weights between the edges  $(i, j)$  and  $(j, i)$  are the same. Finally,  $\mathbf{D} \in \mathbb{R}^{N \times N}$  is the degree matrix of  $G$  defined as a diagonal matrix such that  $\mathbf{D}(i, i) = \sum_{j=1}^N \mathbf{A}(i, j) \forall i \in \mathcal{V}$ .

In our graph, each node is represented by a 853-dimensional vector  $\mathbf{x}_v$  for  $v \in \mathcal{V}$ , and grouped in the matrix  $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_N]^T$ . The feature vector  $\mathbf{x}_v$  is obtained us-



ing a temporal median filter for background initialization, optical flow, intensity, and texture features, after converting the videos to gray-scale. Let  $\mathbf{I}_v^t$  be the Region of Interest (RoI) image of node  $v \in \mathcal{V}$  in the current ( $t$ ) frame. The RoI is the group of pixels belonging to an object instance produced by the instance segmentation method (the bounding box output of Mask R-CNN). Let  $\mathcal{P}_v$  be the set of pixel-indices corresponding to the mask object  $v$  (the mask output of Mask R-CNN). Let  $\mathbf{B}$  be the output image of the temporal median filter and let  $\mathbf{B}_v$  be the RoI image of the background of node  $v$ , *i.e.*, we are extracting the bounding box of  $v$  in  $\mathbf{B}$ . We compute the optical flow vectors, of the current frame with support in  $\mathcal{P}_v$ , and their amplitudes and orientations. Afterward, we construct histograms and their descriptive statistics: minimum, maximum, mean, standard deviation, mean absolute deviation, and range. We also use the RoI images  $\mathbf{I}_v^t$ ,  $\mathbf{I}_v^{t-1}$ ,  $\mathbf{B}_v$  and  $|\mathbf{I}_v^t - \mathbf{B}_v|$  to derive a texture representation, with local binary patterns [35], and the intensity histograms. All the previous features are concatenated into three feature vector  $\mathbf{x}_v$  representing the node  $v$ .

The construction of the undirected graph  $G$  is performed using the k-nearest neighbors algorithm with  $k = 30$  in the matrix  $\mathbf{X}$ . The elements of the adjacency matrix are computed with a Gaussian kernel  $\mathbf{A}(i, j) = \exp(-\frac{d(i, j)^2}{\rho^2})$ , where  $d(i, j) = \|\mathbf{x}_i - \mathbf{x}_j\|$ , and  $\rho$  is the standard deviation of the Gaussian function, set to  $\rho = \frac{1}{|\mathcal{E}|+N} \sum_{(i, j) \in \mathcal{E}} d(i, j)$ . Furthermore, the full set of labels  $\mathbf{Y} \in \mathbb{R}^{N \times F}$  of our algorithm is a matrix encoding the classes of all nodes in  $\mathcal{V}$ , where  $F = 2$  (background and foreground). Each row of  $\mathbf{Y}$  is represented with the so-called one-hot vector: background  $[1, 0]$ , and foreground  $[0, 1]$ .

### 3.3. Graph Semi-Supervised Learning Algorithm

The previous semi-supervised algorithm GraphBGS [16] relies on the sampling and reconstruction of graph signals [11, 37, 2, 38]. As a consequence, Giraldo and Bouwmans [16] defined the sampled nodes (or training set in our case) as a subset of nodes  $\mathcal{S} \subset \mathcal{V}$  with  $\mathcal{S} = \{s_1, s_2, \dots, s_m\}$ , where  $m = |\mathcal{S}| \leq N$  is the number of sampled nodes. GraphBGS applied a variational method in graphs [39] to solve the semi-supervised learning problem, the application of this variational method assumes that the background/foreground nodes are smooth in the graph. Unlike GraphBGS, our algorithm avoids the smoothness assumption by solving the learning problem with GCNNs, and we refer to the set  $\mathcal{S}$  as the training set in GraphMOD-Net.

The layer-wise propagation rule of our GraphMOD-Net, inspired from Kipf and Welling [24] method, is given as follows:

$$\mathbf{H}^{(l+1)} = \sigma(\tilde{\mathbf{D}}^{-\frac{1}{2}} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{-\frac{1}{2}} \mathbf{H}^{(l)} \mathbf{W}^{(l)}) \quad (1)$$

where  $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$  is the adjacency matrix of  $G$  with added

self-connections<sup>1</sup>,  $\mathbf{I}$  is the identity matrix,  $\tilde{\mathbf{D}}$  is the degree matrix of  $\tilde{\mathbf{A}}$ ,  $\mathbf{W}^{(l)}$  is the matrix of trainable weights in layer  $l$ ,  $\sigma(\cdot)$  denotes an activation function, and  $\mathbf{H}^{(l)}$  is the matrix of activations in layer  $l$  such that  $\mathbf{H}^{(0)} = \mathbf{X}$  is the input matrix denoting the representation of the nodes. The propagation rule in Eqn. (1) is motivated by the first-order approximation of localized spectral filters on graphs [20, 12]. For further details, the reader is referred to Kipf and Welling paper [24].

GraphMOD-Net uses a GCNN with one hidden layer to solve the semi-supervised learning problem. The forward of our model is computed using the propagation rule of Eqn. (1) as follows:

$$\mathbf{Z} = f(\mathbf{X}, \mathbf{A}) = \text{softmax} \left( \tilde{\mathbf{D}}^{-\frac{1}{2}} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{-\frac{1}{2}} \text{ReLU} \left( \tilde{\mathbf{D}}^{-\frac{1}{2}} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{-\frac{1}{2}} \mathbf{X} \mathbf{W}^{(0)} \right) \mathbf{W}^{(1)} \right), \quad (2)$$

where  $\mathbf{Z}$  is the output of the GCNN;  $\mathbf{W}^{(0)} \in \mathbb{R}^{C \times H}$  is the weights of the first hidden layer such that  $C$  is the dimension of vector  $\mathbf{x}_v$ , *i.e.*,  $C = 853$ , and  $H$  is the number of feature maps;  $\mathbf{W}^{(1)} \in \mathbb{R}^{H \times F}$  is the weights of the output layer such that  $F$  is the number of classes (*i.e.*,  $F = 2$ : background and foreground);  $\text{ReLU}(\cdot) = \max(0, \cdot)$  is the rectified linear unit; and finally the softmax activation function is defined as  $\text{softmax}(x_i) = \frac{1}{S} \exp(x_i)$  where  $S = \sum_i \exp(x_i)$ . Here the softmax function is applied row-wise. We set the number of feature maps  $H$  in the hidden layer to 16 in our model. The loss function of this GCNN is defined as the cross-entropy error over all labeled nodes:

$$\mathcal{L} = - \sum_{i \in \mathcal{S}} \sum_{j=1}^F \mathbf{Y}(i, j) \ln \mathbf{Z}(i, j). \quad (3)$$

We use the Adam optimizer [23] for training and set the training batch to the full training set in each iteration (this is possible since the whole database fits in memory).

## 4. Experimentation

This section introduces the databases used in this paper, the experimentation protocol, the evaluation metrics, the experiments, and the implementation details of GraphMOD-Net.

### 4.1. Databases

We use the Change Detection 2014 (CDNet2014) [49] and the UCSD background subtraction [31] datasets in this work. CDNet2014 contains 11 challenges including, bad weather, low frame rate, night videos, PTZ, turbulence, baseline, dynamic background, camera jitter, intermittent

<sup>1</sup>The self-connections given by  $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$  were introduced by Kipf and Welling [24] to avoid numerical instabilities and exploding/vanishing gradients.

object motion, shadow, and thermal. Each challenge has between 4 to 6 videos, and each video has from 600 up to 7999 images. Certain frames in each video contain ground-truth images, consisting of pixel-level annotations of foreground and background. On the other hand, UCSD contains 18 videos mainly composed of moving camera sequences, with 30 up to 246 frames each sequence. Each video of UCSD is partially or fully annotated with pixel-level ground-truth images of foreground and background.

## 4.2. Experimentation protocol

The selection of the training, validation, and test nodes is based on an agnostic-video evaluation methodology in order to best evaluate the performance of GraphMOD-Net on unseen videos. To that end, the testing nodes should come from different videos with respect to the training nodes. Formally, let  $\mathcal{V}_k$  be a partition of the whole set of nodes  $\mathcal{V} \subset \mathcal{V}$ , and let  $\mathcal{U}_k \subset \mathcal{V}_k$  be the set of nodes corresponding to the unseen videos for that specific partition  $k$ . Then, the training set  $\mathcal{S} \subset \mathcal{V}_k$  is randomly chosen from a subset of possible nodes  $\mathcal{V}_k \setminus \mathcal{U}_k$ . In other words, the training set is  $\mathcal{S}$ , the test set is  $\mathcal{U}_k$ , and the validation set  $\mathcal{T}_k$  is randomly chosen from the subset  $\mathcal{T}_k \subset \mathcal{V}_k \setminus (\mathcal{S} \cup \mathcal{U}_k)$ .

For CDNet2014 we adopt the same partition of the database as Tezcan *et al.* [48]. The database is divided in 18 sets, *i.e.*,  $k = \{1, \dots, 18\}$ , where each split contains a list of training and test sequences. GraphMOD-Net randomly selects a small subset of nodes  $\mathcal{S} \subset \mathcal{V}_k$  and a small set  $\mathcal{T}_k$  for validation from the training sequences to train and then evaluate the GCNN on unseen videos. For example, the first partition is given such that  $\mathcal{V}_1$  contains the nodes corresponding to the challenges: baseline, bad weather, intermittent object motion, low framerate, and shadow; while the set of unseen videos  $\mathcal{U}_1$  contains the sequences: office, and PETS2006 from baseline; and backdoor, copy machine, and people in shade from shadow. The cardinality of  $\mathcal{T}_k$  is fixed for all experiments such that  $|\mathcal{T}_k| = N \times 0.01$ , *i.e.*, we are using 1% of the dataset as validation. The 18 partitions for the CDNet2014 is given in the supplementary material of Tezcan *et al.* paper [48].

For UCSD we train our GraphMOD-Net in a subset of nodes of CDNet2014 and we test in UCSD dataset. For the training and validation sets we randomly choose nodes from the challenges baseline, dynamic background, shadow, and PTZ of CDNet2014.

## 4.3. Evaluation Metrics

GraphMOD-Net is evaluated using the F-measure defined as follows:

$$\text{F-measure} = 2 \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}, \quad (4)$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \text{ Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \quad (5)$$

with TP, FP, and FN the number of True Positives, False Positives, and False Negatives pixels, respectively.

## 4.4. Experiments

For CDNet2014, GraphMOD-Net constructs a graph for the whole dataset, resulting in an undirected graph of 258956 nodes. For each partition of the dataset a percentage of the total amount of nodes in the set  $\mathcal{M} = \{0.001, 0.005, 0.05, 0.1\}$  is used for training and then evaluated with unseen videos. For example,  $\mathcal{S}$  contains 259 nodes for  $0.001 \in \mathcal{M}$  (keep in mind that usually each image contains several nodes as shown in Fig. 2). The training of our GCNN is validated with a Monte Carlo cross-validation with 3 repetitions for each training density and split of the database. As a consequence, we train  $18 \times 3 \times |\mathcal{M}| = 216$  GCNNs for the validation of GraphMOD-Net.

For UCSD, GraphMOD-Net constructs a graph with the whole UCSD dataset plus the challenges baseline, dynamic background, shadow, and PTZ of CDNet2014; resulting in a graph of 104261 nodes. In this case, the test set is the whole UCSD dataset, and a small number of nodes in the set  $\mathcal{M}$  is used for training. A Monte Carlo cross-validation with 5 repetitions is performed for each training density, *i.e.*, we train  $5 \times |\mathcal{M}| = 20$  GCNNs for UCSD dataset. For the sake of clarification, the metrics in the experiments are computed on the unseen set of nodes in each experiment. The CDNet2014 and UCSD experiments follow the guidelines for the training, validation, and testing sets of Section 4.2.

## 4.5. Implementation Details

The Cascade and Mask R-CNN were implemented using Pytorch and Detectron2 [50, 51]. The semi-supervised learning GCNN was implemented using TensorFlow [1]. The GCNN has a dropout rate of 0.5 [45], an initial learning rate of 0.01, a weight decay of  $5 \times 10^{-4}$ , and a maximum number of 600 epochs. We also implemented an early stopping with a window size of 10, *i.e.*, the training is stopped when the validation loss does not decrease for 10 consecutive epochs.

For the comparison with GraphMOD-Net, the background subtraction algorithms MoG [47], SuBSENSE [46], and ViBe [3] for UCSD were implemented with the BGSLibrary [43]; while GRASTA [21] and DECOLOR [53] were implemented using the LRSLibrary [44].

## 5. Results and Discussion

GraphMOD-Net is compared, either in CDNet2014 or UCSD, with the following state-of-the-art algorithms for MOD: MoG [47], GRASTA [21], DECOLOR [53], Graph Cut Difference (GraphCutDiff) [33], ViBe [3], SuBSENSE [46], WisenetMD [25], GraphBGS [16], BSUV-Net [48], and FgSegNet v2 [26]. The results of FgSegNet v2 [26] for

Table 1. Comparison of the average F-measure over nine challenges of the CDNet2014 dataset, namely: bad weather (BWT), baseline (BSL), camera jitter (CJI), dynamic background (DBA), intermittent object motion (IOM), low frame rate (LFR), PTZ, shadow (SHW), and thermal (THL). The best and second best performing methods for each challenge are shown in **red** and **blue**, respectively.

Method	BWT	BSL	CJI	DBA	IOM	LFR	PTZ	SHW	THL	Overall
GraphCutDiff (unsupervised)	<b>0.879</b>	0.715	0.549	0.539	0.402	0.513	0.372	0.723	0.579	0.5857
SuBSENSE (unsupervised)	0.862	0.950	<b>0.815</b>	0.818	0.657	<b>0.645</b>	0.348	0.899	<b>0.817</b>	0.7567
WisenetMD (unsupervised)	0.862	0.949	<b>0.823</b>	<b>0.838</b>	<b>0.726</b>	0.640	0.337	0.898	0.815	<b>0.7653</b>
GraphBGS (semi-supervised)	0.837	0.942	0.700	0.743	0.405	0.558	<b>0.749</b>	<b>0.966</b>	0.729	0.7366
GraphMOD-Net (semi-supervised, <b>ours</b> )	0.839	<b>0.955</b>	0.720	<b>0.851</b>	0.554	0.521	<b>0.770</b>	<b>0.942</b>	0.682	0.7593
BSUV-net (supervised)	<b>0.871</b>	<b>0.969</b>	0.774	0.797	<b>0.750</b>	<b>0.680</b>	0.628	0.923	<b>0.858</b>	<b>0.8056</b>
FgSegNet v2 (supervised)	0.328	0.693	0.427	0.363	0.200	0.248	0.350	0.530	0.604	0.4159

Table 2. Comparison of the F-measure results over the videos of UCSD background subtraction dataset. The best and second best performing methods for each video are shown in **red** and **blue**, respectively. DECOLOR and GRASTA are subspace learning methods.

Sequence	MoG	DECOLOR	ViBe	GRASTA	SuBSENSE	BSUV-Net	GraphBGS	GraphMOD
Birds	0.1427	0.1457	0.3354	0.1320	0.4832	0.2625	<b>0.7495</b>	<b>0.7391</b>
Boats	0.0881	0.2179	0.1854	0.0678	0.4550	0.6621	<b>0.7765</b>	<b>0.8007</b>
Bottle	0.1856	0.4765	0.4512	0.1159	<b>0.6570</b>	0.5039	<b>0.8741</b>	<b>0.8741</b>
Chopper	0.3237	0.6214	0.4930	0.0842	0.6723	0.3020	<b>0.7766</b>	<b>0.7844</b>
Cyclists	0.0915	0.2224	0.1211	0.1243	0.1445	0.4138	<b>0.7417</b>	<b>0.7479</b>
Flock	0.2706	0.2943	0.2306	0.1612	0.2492	0.0025	<b>0.5903</b>	<b>0.5871</b>
Freeway	0.2622	<b>0.5229</b>	0.4002	0.0814	<b>0.5518</b>	0.1185	0.3719	0.3516
Hockey	0.3867	0.3449	0.4195	0.3149	0.3611	0.6908	<b>0.7664</b>	<b>0.7804</b>
Jump	0.2679	0.3135	0.2636	0.4175	0.2295	<b>0.8697</b>	<b>0.7734</b>	0.7602
Landing	0.0335	0.0640	0.0433	0.0414	0.0026	0.0012	<b>0.2452</b>	<b>0.1840</b>
Ocean	0.1113	0.1315	0.1648	0.1144	0.2533	<b>0.5335</b>	<b>0.8593</b>	<b>0.8593</b>
Peds	0.3731	0.7942	0.5257	0.4653	0.5154	0.6738	<b>0.8518</b>	<b>0.8512</b>
Skiing	0.2038	0.3473	0.1441	0.0927	0.2482	0.0602	<b>0.5963</b>	<b>0.5953</b>
Surf	0.0489	0.0647	0.0462	0.0523	0.0467	0	<b>0.5851</b>	<b>0.6139</b>
Surfers	0.0542	0.1959	0.1189	0.0742	0.1393	0.4776	<b>0.6719</b>	<b>0.6611</b>
Traffic	0.2188	<b>0.2732</b>	0.1445	0.0368	0.1165	0	<b>0.5722</b>	<b>0.5722</b>
Overall	0.1914	0.3144	0.2555	0.1485	0.3203	0.3483	<b>0.6751</b>	<b>0.6727</b>

CDNet2014 are reported using unseen videos for the evaluation of the network, the performance comes from Tezcan *et al.* paper [48].

Tables 1 and 2 show the comparison of GraphMOD-Net with several state-of-the-art methods in CDNet2014 and UCSD, respectively. The results of GraphMOD-Net and GraphBGS are computed with the best F-measures in each sequence from the experiments (both algorithms have the same instance segmentation method). For CDNet2014 in Table 1, GraphMOD-Net shows the best results in the challenges baseline, dynamic background, PTZ, and shadow. The results also show that our algorithm has an overall increment in performance with respect to GraphBGS. For UCSD in Table 2, GraphMOD-Net show either the best or the second-best results across almost all the videos of UCSD. In the overall performance, GraphMOD-Net comes second lagging slightly behind GraphBGS. We believe it is a consequence of increasing the representational power when solving the semi-supervised learning problem with GCNNs leading to a slightly worse generalization. Similarly, GraphMOD-Net outperforms subspace learning methods such as DECOLOR and GRASTA. Subspace learning methods generally have problems when dealing with moving camera sequences.

We can also notice the degradation of the deep learning method FgSegNet v2 compared to the original results [26] when tested on unseen videos in Table 1. Similarly, the performance of BSUV-Net drops when applied on UCSD, these results could suggest a weakness of deep

learning methods in generalization for MOD (keep in mind that BSUV-Net uses the output of a fully CNN trained in ADE20K dataset [52] as input of the network). In Table 2, we evaluated the BSUV-Net model provided by the authors of [48] that was trained in the CDNet2014. In our case, we trained GraphMOD-Net in a small subset of nodes of CDNet2014 including the challenges baseline, dynamic background, shadow, and PTZ. This suggests that GraphMOD-Net generalizes better than BSUV-Net when applied on a different dataset from which it was trained. For the sake of clarification, we did not use the ground truth of UCSD in the training procedure.

Figure 3 depicts the results of our GraphMOD-Net compared with the previous semi-supervised MOD method GraphBGS [16]. The results of the challenges turbulence and night videos are not displayed since our Mask R-CNN fails to detect several instances in these videos. As a result, it is not possible to detect the moving objects in some sequences for the GCNN. GraphMOD-Net is better than GraphBGS in the challenges of bad weather, baseline, camera jitter, dynamic background, intermittent object motion, low frame rate, and PTZ; while GraphBGS shows better results than GraphMOD-Net in the challenges shadow and thermal. GraphMOD-Net benefits from the higher modeling capacity of GCNNs by improving upon the GraphBGS as shown in Tables 1, 2, and in Figure 3.

Table 3 shows some qualitative results of GraphMOD-Net compared with BSUV-Net [48], and GraphBGS [16] for CDNet2014. In the same way, Table 4 shows visual results

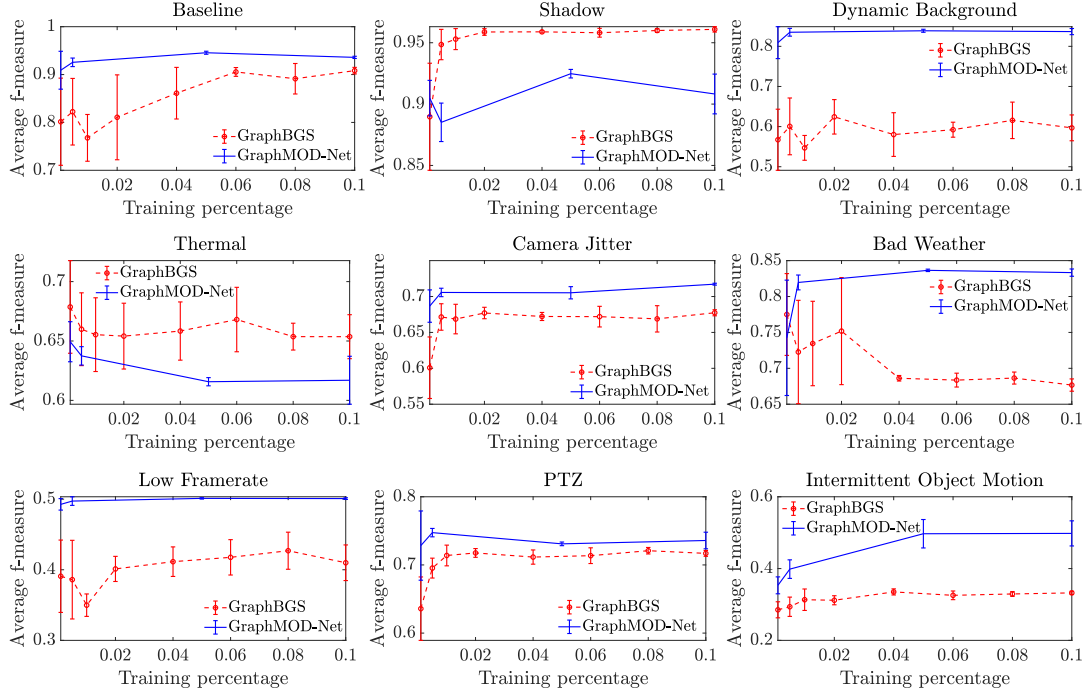


Figure 3. Average F-measure vs training percentage in the nine challenges of CDNet2014, using GraphBGS, and our GraphMOD-Net. Each point in the plots is a Monte Carlo cross-validation experiment with 3 repetitions.
























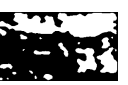





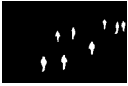
Table 3. Some visual results of GraphMOD-Net in CDNet2014 compared with state-of-the-art algorithms, from left to right: original images, ground-truth images, BSUV-Net [48], GraphBGS [16], and the proposed GraphMOD-Net.

Categories	Original	Ground Truth	BSUV-Net	GraphBGS	GraphMOD-Net
Bad Weather wetSnow in000500					
Baseline Pedestrians in000496					
Camera Jitter Boulevard in000915					
Dynamic Back. Fall in002789					
PTZ Two Position. in002789					
Low-F Rate TramCross in000607					

of GraphMOD-Net on UCSD compared with DECOLOR and SuSENSE. For a fair comparison, the results in Table 3 for GraphBGS and GraphMOD-Net are computed with the same training percentage on each sequence. The visual results of GraphMOD-Net indicate that GCNNs are better than previous unsupervised and supervised algorithms

used for MOD in some challenges. The improvement of GraphMOD-Net with respect to GraphBGS is because of the modeling capacity of GCNNs compared with previous semi-supervised learning methods, where a smooth prior assumption is required.

Table 4. Some visual results of GraphMOD-Net in UCSD compared with state-of-the-art algorithms, from left to right: original images, ground-truth images, DECOLOR [53], SuBSENSE [46], and the proposed GraphMOD-Net.

Videos	Original	Ground Truth	DECOLOR	SuBSENSE	GraphMOD-Net
Birds frame_4					
Boats frame_12					
Bottle frame_14					
Chopper frame_47					
Cyclists frame_9					
Peds frame_25					

## 6. Conclusions

This paper introduces a method for MOD based on GCNN. The pipeline of the method involves a Cascade Mask R-CNN or a Mask R-CNN to get the object instances from the videos; a graph construction with k-nearest neighbors, where nodes are associated to object instances, and represented with feature vector encompassing optical flow, intensity and texture descriptors. We use a compact GCNN model trained in a semi-supervised mode. Our algorithm outperforms previous state-of-the-art unsupervised, semi-supervised, and supervised learning algorithms for MOD in several challenges of the CDNet2014 and UCSD datasets. In the future, we will further study the implications of the architecture of GCNNs in the problem of MOD. In the same way, we will analyze some theoretical aspects in the training of the GCNNs with focus in MOD. And finally, we will study a inductive framework [19] for GraphMOD-Net in the context of MOD.

## References

- [1] M. Abadi et al. Tensorflow: A system for large-scale machine learning. In *Symposium on Operating Systems Design and Implementation*, 2016. 5
- [2] A. Anis et al. A sampling theory perspective of graph-based semi-supervised learning. *IEEE T-IT*, 65(4):2322–2342, 2018. 4
- [3] O. Barnich and M. Van Droogenbroeck. ViBe: A universal background subtraction algorithm for video sequences. *IEEE T-IP*, 20(6):1709–1724, 2010. 5
- [4] T. Bouwmans. Subspace learning for background modeling: A survey. *Recent Patents on Computer Science*, 2(3):223–234, 2009. 2
- [5] T. Bouwmans. Traditional and recent approaches in background modeling for foreground detection: An overview. *Computer science review*, 11:31–66, 2014. 1, 2
- [6] T. Bouwmans, F. El Baf, and B. Vachon. Background modeling using mixture of Gaussians for foreground detection-a survey. *Recent Patents on Computer Science*, 1(3):219–237, 2008. 2
- [7] T. Bouwmans et al. Decomposition into low-rank plus additive matrices for background/foreground separation: A review for a comparative evaluation with a large-scale dataset. *Computer Science Review*, 23:1–71, 2017. 1
- [8] T. Bouwmans et al. Deep neural network concepts for background subtraction: A systematic review and comparative evaluation. *Neural Networks*, 2019. 2
- [9] T. Bouwmans and E. H. Zahzah. Robust PCA via principal component pursuit: A review for a comparative evaluation in video surveillance. *CVIU*, 122:22–34, 2014. 2
- [10] Z. Cai and N. Vasconcelos. Cascade R-CNN: High quality object detection and instance segmentation. *IEEE T-PAMI*, 2019. 1, 3
- [11] S. Chen et al. Discrete signal processing on graphs: Sampling theory. *IEEE T-SP*, 63(24):6510–6523, 2015. 2, 4
- [12] M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *NeurIPS*, 2016. 4
- [13] S. S. Du et al. How many samples are needed to estimate a convolutional neural network? In *NeurIPS*, 2018. 2



- [14] B. Garcia-Garcia, T. Bouwmans, and A. J. Rosales. Background subtraction in real applications: Challenges, current models and future directions. *Computer Science Review*, 35, 2020. 1
- [15] J. García-González et al. Foreground detection by probabilistic modeling of the features discovered by stacked denoising autoencoders in noisy video sequences. *PRL*, 125:481–487, 2019. 2
- [16] J. H. Giraldo and T. Bouwmans. GraphBGS: Background subtraction via recovery of graph signals. In *ICPR*, 2021. 2, 3, 4, 5, 6, 7
- [17] J. H. Giraldo, S. Javed, and T. Bouwmans. Graph moving object segmentation. *IEEE T-PAMI*, early access, Dec. 9, 2020. 1, 3
- [18] J. H. Giraldo, A. Salazar, A. Gomez, and A. Diaz-Pulido. Camera-trap images segmentation using multi-layer robust principal component analysis. *The Visual Computer*, 35(3):335–347, 2019. 1
- [19] W. Hamilton, Z. Ying, and J. Leskovec. Inductive representation learning on large graphs. In *NeurIPS*, 2017. 8
- [20] D. K. Hammond, P. Vandergheynst, and R. Gribonval. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 30(2):129–150, 2011. 4
- [21] J. He, L. Balzano, and A. Szlam. Incremental gradient on the grassmannian for online foreground and background separation in subsampled video. In *CVPR*, 2012. 5
- [22] K. He et al. Mask R-CNN. In *ICCV*, 2017. 1, 3
- [23] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 4
- [24] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017. 1, 4
- [25] S. H. Lee, G. C. Lee, J. Yoo, and S. Kwon. WisenetMD: motion detection using dynamic background region analysis. *Symmetry*, 11(5):621, 2019. 5
- [26] L. A. Lim and H. Y. Keles. Learning multi-scale features for foreground segmentation. *PAA*, 23(3):1369–1380, 2020. 2, 5, 6
- [27] T. Y. Lin et al. Microsoft COCO: Common objects in context. In *ECCV*, 2014. 3
- [28] T. Y. Lin et al. Feature pyramid networks for object detection. In *CVPR*, 2017. 3
- [29] K. S. Lu and A. Ortega. Fast graph Fourier transforms based on graph symmetry and bipartition. *IEEE T-SP*, 67(18):4855–4869, 2019. 3
- [30] B. D. Lucas, T. Kanade, et al. An iterative image registration technique with an application to stereo vision. 1981. 1, 3
- [31] V. Mahadevan and N. Vasconcelos. Spatiotemporal saliency in dynamic scenes. *IEEE T-PAMI*, 32(1):171–177, 2009. 1, 2, 4
- [32] T. Minematsu et al. Analytics of deep neural network-based background subtraction. *Journal of Imaging*, 4(6):78, 2018. 2
- [33] A. Miron and A. Badii. Change detection based on graph cuts. In *IWSSIP*, 2015. 5
- [34] M. Narayana, A. Hanson, and E. G. Learned-Miller. Background subtraction: separating the modeling and the inference. *MVA*, 25(5):1163–1174, 2014. 2
- [35] T. Ojala, M. Pietikäinen, and T. Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE T-PAMI*, (7):971–987, 2002. 1, 4
- [36] O. Oreifej, X. Li, and M. Shah. Simultaneous video stabilization and moving object detection in turbulence. *IEEE T-PAMI*, 35(2):450–462, 2012. 1
- [37] A. Ortega et al. Graph signal processing: Overview, challenges, and applications. *Proceedings of the IEEE*, 106(5):808–828, 2018. 2, 4
- [38] A. Parada-Mayorga et al. Blue-noise sampling on graphs. *IEEE T-SIPN*, 5(3):554–569, 2019. 3, 4
- [39] I. Pesenson. Variational splines and Paley-Wiener spaces on combinatorial graphs. *Constructive Approximation*, 29(1):1–21, 2009. 2, 4
- [40] D. S. Pham, O. Arandjelović, and S. Venkatesh. Detection of dynamic background due to swaying movements from motion features. *IEEE T-IP*, 24(1):332–344, 2014. 1
- [41] P. Rodriguez and B. Wohlberg. Incremental principal component pursuit for video background modeling. *Journal of Mathematical Imaging and Vision*, 55(1):1–18, 2016. 2
- [42] O. Russakovsky et al. Imagenet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015. 2
- [43] A. Sobral and T. Bouwmans. BGS library: A library framework for algorithm’s evaluation in foreground/background segmentation. In *Handbook on "Background Modeling and Foreground Detection for Video Surveillance"*, Chapter 23. 2014. 5
- [44] A. Sobral, T. Bouwmans, and E. Zahzah. LRSLibrary: Low-rank and sparse tools for background modeling and subtraction in videos. In *Robust Low-Rank and Sparse Matrix Decomposition: Applications in Image and Video Processing*. CRC Press, Taylor and Francis Group, 2015. 5
- [45] N. Srivastava et al. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 15(1):1929–1958, 2014. 5
- [46] P. L. St-Charles, G. A. Bilodeau, and R. Bergevin. SuB-SENSE: A universal change detection method with local adaptive sensitivity. *IEEE T-IP*, 24(1):359–373, 2014. 1, 2, 5, 8
- [47] C. Stauffer and W. E. L. Grimson. Adaptive background mixture models for real-time tracking. In *CVPR*, 1999. 5
- [48] O. Tezcan, P. Ishwar, and J. Konrad. BSUV-Net: A fully-convolutional neural network for background subtraction of unseen videos. In *WACV*, 2020. 1, 2, 5, 6, 7
- [49] Y. Wang et al. CDnet 2014: An expanded change detection benchmark dataset. In *CVPR*, 2014. 1, 2, 4
- [50] Y. Wu et al. Detectron2. <https://github.com/facebookresearch/detectron2>, 2019. 5
- [51] H. Zhang et al. ResNeSt: Split-attention networks. *arXiv preprint arXiv:2004.08955*, 2020. 3, 5
- [52] B. Zhou et al. Scene parsing through ADE20K dataset. In *CVPR*, 2017. 6
- [53] X. Zhou, C. Yang, and W. Yu. Moving object detection by detecting contiguous outliers in the low-rank representation. *IEEE T-PAMI*, 35(3):597–610, 2012. 5, 8
- [54] J. Zuo et al. Moving target detection based on improved gaussian mixture background subtraction in video images. *IEEE Access*, 7:152612–152623, 2019. 2